March 2021

Triggering Applications From A Distance Using Patterns Displayed On A Wearable Device

N/A

Triggering Applications From A Distance Using Patterns Displayed On A Wearable Device

ABSTRACT

Currently, voice interaction between users and smart devices does not use spatial context, e.g., the position of the user relative to the smart device, in responding to user queries which limits the possibilities of interaction and the precision of the answers provided by the smart device. This disclosure describes techniques that enable a user to trigger an application on a smart device such as a smart display or smart speaker by flashing a predefined pattern on a wearable device such as a smartwatch worn by the user. The predefined pattern is detected and utilized to determine the distance to the user and optionally, to map to specific commands. An application is triggered on the smart device based on the distance, with contextually appropriate settings. By enabling fast, spatially-aware, high-resolution, gesture-based interfaces, the described techniques can complement, and, in some cases supplant, time-consuming voice interfaces.

KEYWORDS

- Smart device
- Smart speaker
- Smartwatch
- Display pattern
- Distance-based application triggering
- Spatially-aware response
- Virtual assistant
- Gesture recognition
- Pose estimation
- Skeletal modeling

BACKGROUND

Smart devices are ubiquitous to the point that they have effectively become ambient, always available assistant devices. For example, a user can issue a simple voice query to retrieve relevant content from a nearby smart speaker or smartphone. However, voice interaction does not currently use spatial context, e.g., the smart device does not leverage the position of the user in a home or other space in answering their query, thereby limiting the possibilities of interaction and the precision of the answers provided.

DESCRIPTION

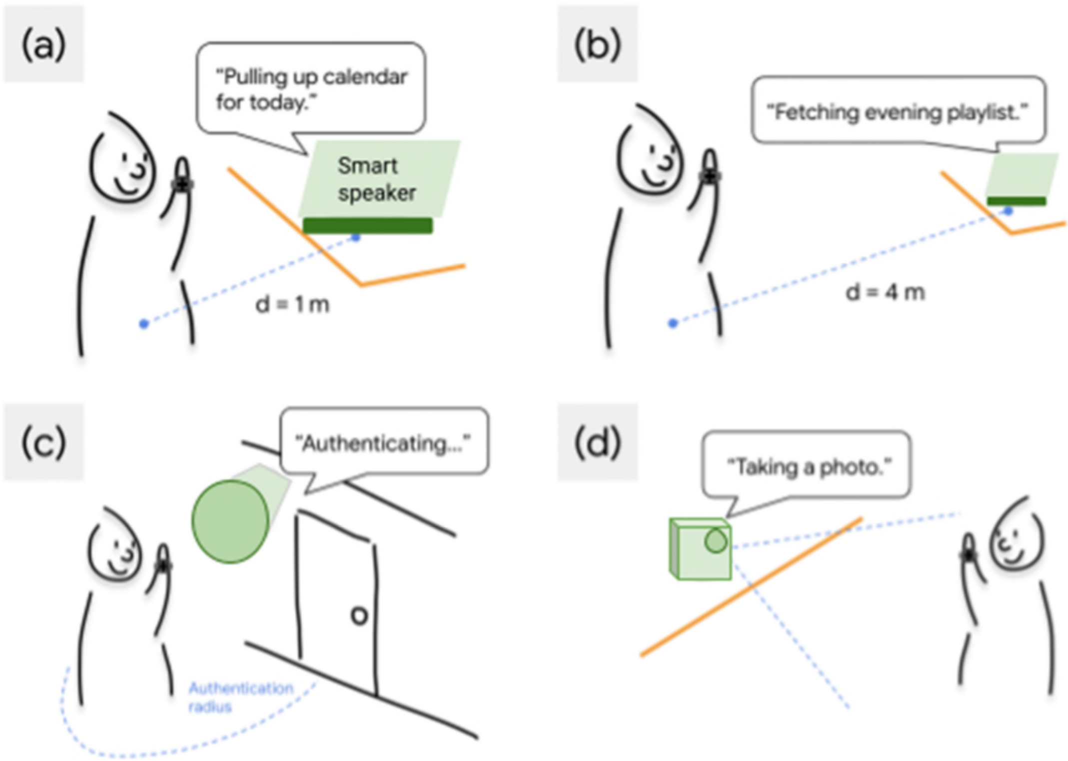


Fig. 1: Triggering applications using smartwatch display patterns. Raising a smartwatch (a) at short distances triggers a calendar on a nearby smart speaker; (b) at longer distances triggers music playback on the smart speaker; (c) within a certain authentication radius from a security camera authenticates a user; (d) within a certain radius from a camera triggers the taking of a photo.

This disclosure describes techniques to enable a user to trigger an application in a smart device, e.g., smart camera, smart speaker, etc., by flashing a predefined pattern on the user's smartwatch (or other wearable device). The predefined pattern is used by the smart device to determine the distance to the user and accordingly trigger distance-specific applications.

For example, per the techniques, a smart device can surface applications based on whether the user triggers the device from the kitchen or from the study room. Some examples of distance-based triggering of applications are illustrated in Fig. 1. Detection of a pattern on the smartwatch when the user is proximate to the smart speaker triggers the smart speaker pulling up the user's calendar (Fig. 1a) while detecting a pattern when the user is further away triggers music playback (Fig. 1b). Similarly, a camera detecting a user displaying a pattern close to it can perform authentication (Fig. 1c), while a camera that is further away can initiate capture of a photo (Fig. 1d). The mapping between distance and triggered application is configurable by the user. Multiple pre-configured patterns can be set up by the user, each corresponding to a certain action to be performed by the smart device.

The size of the fixed pattern is inversely proportional to the user's distance from the device that uses its camera to detect the pattern. The user's coordinates can be determined in all three dimensions, which enables the surfacing of a menu of spatially-aware content. Effectively, the use of the pattern and distance from the detecting device enable a spatially-aware shortcut for fast application triggering by using a smartwatch or other wearable device.

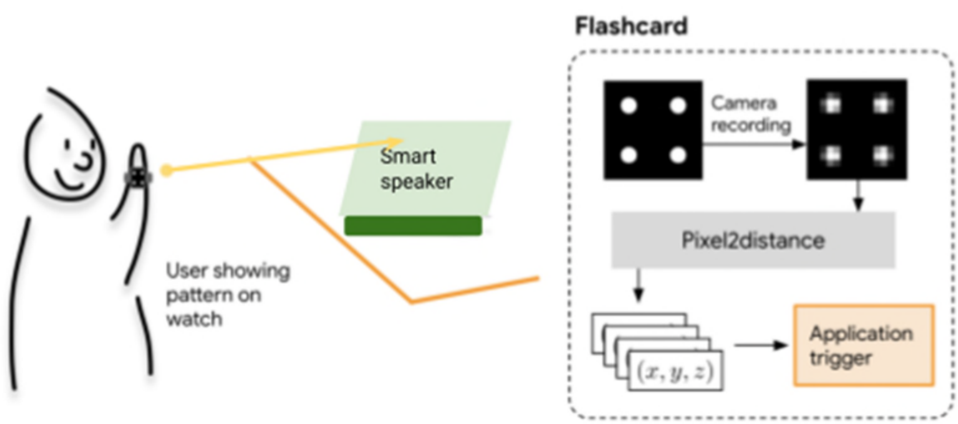


Fig. 2: Determining application to run based on a distance derived from smartwatch pattern

The synergy between a camera-enabled smart device (e.g., a smart speaker, smart display, smartphone, security camera/ other camera, etc.) and a smartwatch (or other wearable device) is leveraged through the following sequence of events, illustrated in Fig. 2:

1. *Raise smartwatch*: The user holds up the smartwatch with its display facing towards the smart device that includes a camera.
2. *Raise-to-flash*: The display shows a pre-configured pattern. An example pattern can be four corner dots for identification.
3. *Pixel-to-distance*: The camera on the smart device detects the unique pattern.
4. *Distance scaling*: Further analytics is done on the pattern to determine user coordinates in (x, y, z).
5. *Application triggering*: The application is triggered based on the detected user coordinates.

The events are described in greater detail below.

<u>*Raise-to-flash*</u>

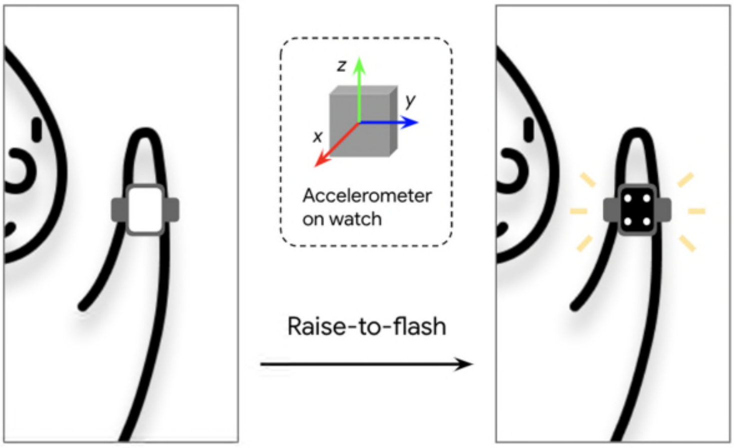


Fig. 3: When the user holds up their watch, the watch display automatically turns into a pre-configured pattern

Raise-to-flash, illustrated in Fig. 3, is a procedure by which the user holds up the watch to change its display automatically into a pre-configured pattern. The action of the user raising the watch and holding it towards the smart device has an identifiable motion trajectory followed by a static coordinate. The motion trajectory can be detected by the inertial measurement unit (IMU), e.g., accelerometer, within the smartwatch.

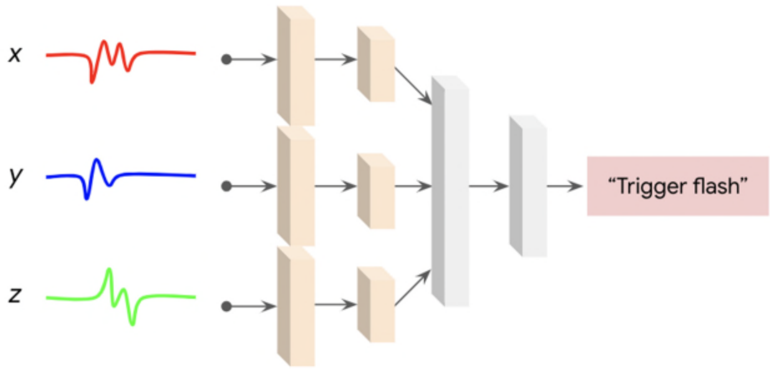


Fig. 4: Accelerometer readings used to determine that the user has raised his watch

As illustrated in Fig. 4, accelerometer readings (x, y, z) can be used to determine that the user has raised their watch, for example, using a multi-header convolutional neural network. The accelerometer captures a time-series of the three-dimensional acceleration vector, which is stored in a 3 x *n* matrix, where *n* is buffer length, spanning, e.g., 0.5 seconds, large enough to capture the raised-hand motion and small enough to deliver a low-latency decision. This matrix is fed into the multi-header convolutional neural network model to deliver a decision on whether the user has raised the smartwatch or not.

Pixel-to-distance

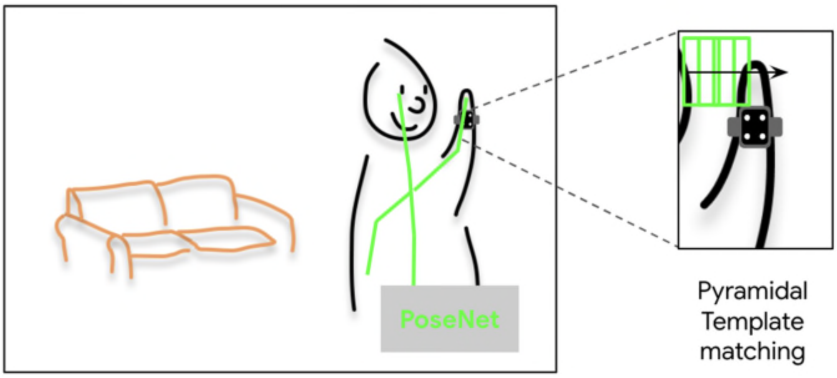


Fig. 5: View from the camera of the smart device

Fig. 5 illustrates the view from the camera of the smart device. The pattern in the scene can be detected by stacking computer vision techniques as follows:

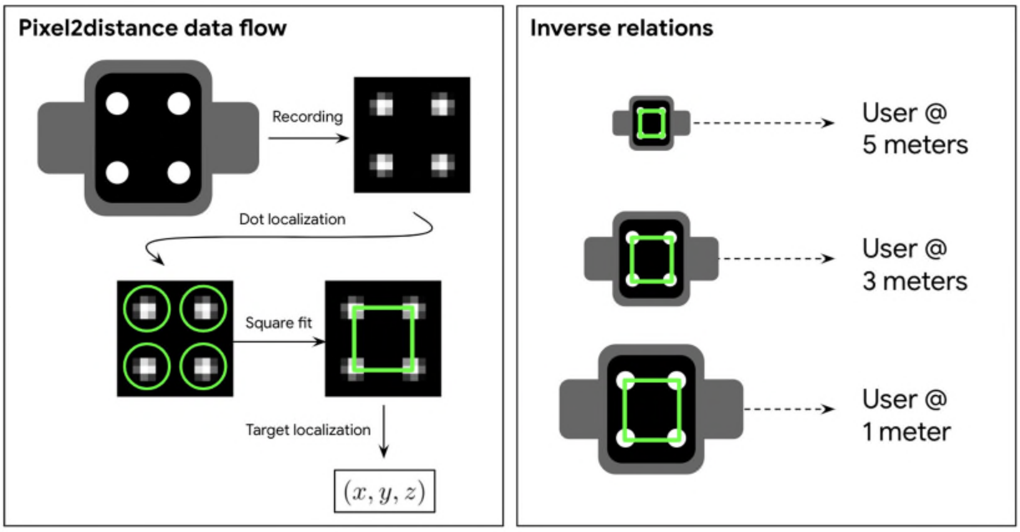
1. An N-point skeletal model of the person in the field of view is constructed using neural network techniques.
2. Pyramidal template matching: From the skeletal model, the coordinates of the human elbow are focused upon, and a determination that it is raised is made with simple heuristics, e.g., checking the angle of the arm. The relevant portion of the image is cropped out and a pyramidal filter is run to find a match with the pre-configured pattern. If this filter score is higher than a threshold, the pattern is confirmed to be detected and its image space coordinates saved. The pyramidal filter is described by the mathematical equations below.

$$\tilde{\mathbf{X}} = \mathbf{X}/\max(\mathbf{X})$$

$$\underset{k \in \{1,\ldots,M\}}{\text{maximize}} \sum_{i=1}^{N} \sum_{j=1}^{N} \left(\mathbf{A}_{i,j}^{(k)} \cdot \tilde{\mathbf{X}}_{i,j} \right)$$

In the above, **X** is the cropped out portion of the image where the pre-configured pattern is believed to be. $\mathbf{A}^{(k)}$ is a series of matching templates for the pre-configured pattern. For example, $\mathbf{A}^{(1)}$ can be the pre-configured pattern when observed at a distance of 0.5 meters, $\mathbf{A}^{(2)}$ the pre-configured pattern when observed at a distance of 1.0 meter, etc. The indices i and j are pixel coordinates, and N is the length (or breadth) in pixels of the cropped image. The distance of the user is determined as that k which maximizes the inner product (correlation) between the template $\mathbf{A}^{(k)}$ and the image **X**.

3. Distance scaling: Once the pattern is detected and localized in the image space, a dot detector is run to identify the four corners. A square-fitting algorithm is run on the four identified coordinates. The distance is inferred from the pixel width and height of the square. As mentioned earlier, distance is inversely correlated with square size.



(a) **(b)**

Fig. 6: (a) The pixel-to-distance workflow; (b) An inverse relationship between the size of the pattern and the distance between the camera and the user

The pixel-to-distance workflow, e.g., recording (capturing) the pattern, localizing the dots, square-fitting, target localization, etc., is illustrated in Fig. 6(a). Fig. 6(b) illustrates the inverse relationship between the size of the pattern and the distance between the user and the camera.

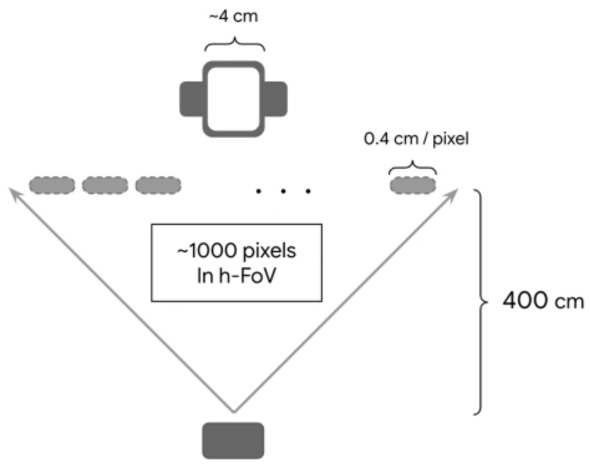


Fig. 7: Accurate ranging for typical smartwatch displays and camera sizes

For typical smartwatch displays, e.g., of size 4 cm, and camera resolutions, e.g., of length (or breadth) 1000 pixels for a one megapixel camera, a device-to-person distance of 4 meters can be ranged. At a distance of 4 meters, the number of image pixels that can fit into a 4 cm width is about 10; beyond 4 meters, the number of pixels that can fit into the watch display may be too few for accurate ranging.

As illustrated in Fig. 8, once the coordinates are identified, an extra layer of in-air gestural interaction, also spatially tagged, can be enabled using the IMU onboard the smartwatch.

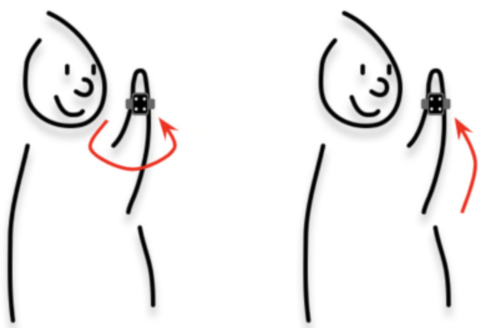


Fig. 8: Additional spatially-tagged gestural interactions

For example, when the user is in the kitchen (defined by a particular coordinate system), a circle hand gesture can be for reviewing a recipe and the raised-arm gesture can be for flipping through recipe pages. When the user is in the living room, the circle gesture can be mapped to a “skip to next media” command and the raised-arm gesture can be mapped to an “increase volume” command. By enabling this fast, spatially-aware, high-resolution, gesture-based interface, the described techniques can complement, and, in some cases supplant, the potentially time-consuming voice interface.

Further to the descriptions above, a user is provided with controls allowing the user to make an election as to both if and when systems, programs, or features described herein may enable the collection of user information (e.g., information about a user’s smart devices, a user’s gestures, a user’s preferences, or a user’s current location), and if the user is sent content or communications from a server. In addition, certain data are treated in one or more ways before it is stored or used, so that personally identifiable information is removed. For example, a user’s identity is treated so that no personally identifiable information can be determined for the user. Thus, the user has control over what information is collected about the user, how that information is used, and what information is provided to the user.

CONCLUSION

Currently, voice interaction between users and smart devices does not use spatial context, e.g., the position of the user relative to the smart device, in responding to user queries which limits the possibilities of interaction and the precision of the answers provided by the smart device. This disclosure describes techniques that enable a user to trigger an application on a smart device such as a smart display or smart speaker by flashing a predefined pattern on a wearable device such as a smartwatch worn by the user. The predefined pattern is detected and utilized to determine the distance to the user and optionally, to map to specific commands. An application is triggered on the smart device based on the distance, with contextually appropriate settings. By enabling fast, spatially-aware, high-resolution, gesture-based interfaces, the described techniques can complement, and, in some cases supplant, time-consuming voice interfaces.

REFERENCES

[1] Bâce, Mihai, Sander Staal, Gábor Sörös, and Giorgio Corbellini. "Collocated Multi-user Gestural Interactions with Unmodified Wearable Devices." *Augmented Human Research* 2, no. 1 (2017): 1-14.

[2] Darbar, Rajkumar, Prasanta Kr Sen, Punyashlok Dash, and Debasis Samanta. "Using hall effect sensors for 3D space text entry on smartwatches." *Procedia computer science* 84 (2016): 79-85.

[3] Real-time Human Pose Estimation in the Browser with TensorFlow.js, available online at https://medium.com/tensorflow/real-time-human-pose-estimation-in-the-browser-with-tensorflow-js-7dd0bc881cd5